CS 46B

Introduction to Data Structures

Summer Semester 2015

Department of Computer Science
San José State University
Instructor: Ron Mak

Homework #10
Huffman Coding

Assigned:	Thursday, July 30
Final due:	Wednesday, August 5 at 11:59 PM
Codecheck URL:	http://codecheck.it/codecheck/files/1508010323ehsg28urjg6q8sjsian3rrpmi
Canvas:	Homework 10 Final
Points:	20 points max

This assignment will help you understand the Huffman binary tree and Huffman coding.

You will generate a Huffman tree using the characters from the input text file **GettysburgAddress.txt** and their frequencies. If the text file was originally encoded using the 8-bit UTF code, how much space savings will there be if the file is encoded using the Huffman code?

You will use **HuffmanTree.java** that you can download from your textbook's website:
http://bcs.wiley.com/he-bcs/Books?action=index&itemId=1118431111&bcsId=7872

Write a new class **`HuffmanGettysburg`** that has at least the following methods:

- Method **`makeFrequencyMap()`** reads the input file and returns a frequency map that maps each character in the file to its frequency.

```
private static Map<Character, Integer> makeFrequencyMap(Scanner in)
```

- Method **`computeCharacterCount()`** iterates over the frequency map and returns the total number of characters in the input file.

```
private static int computeCharacterCount(Map<Character, Integer> map)
```

- Method **`printHuffmanCodes()`** uses both the frequency map and the encoding map (obtained from the Huffman tree) to print a table of the characters in the input file, one character per row in sorted order. For each character, print the character, its frequency, and its Huffman code. The method should also return the total bit length of the file if all the characters are encoded using their Huffman codes.

```
private static int printHuffmanCodes(Map<Character, Integer> frequencyMap,
                                     Map<Character, String>  encodingMap)
```

- The **`main()`** method should:
    - Create a file **`Scanner`** for the input text file **GettysburgAddress.txt**. (The input file is already loaded into Codecheck.)
    - Create a Huffman tree and its encoding map.
    - Call the above methods appropriately.
    - Print the total number of characters and the total number of bits required to encode the file using UTF-8 encoding (the way text files are normally encoded).
    - Print the total number of bits required to encode all the input file's characters using their Huffman codes.
    - Print the percentage reduction in bit length of the input file from using the UTF-8 code to using the Huffman code.

Note that you are not expected to actually encode the input file using Huffman coding. Just calculate how many bits it would take to do the encoding.

Expected output

```
 1459 total characters
11672 total UTF-8 bits

Character Frequency Code
    \n        25     101010
 space       244     111
     ,        23     100001
     -         8     11001011
     .        10     1010000
     B         1     0010010000
     F         1     0010010011
     G         1     0010010001
     I         3     001001011
     N         1     0010010010
     T         2     1100101001
     W         2     001001010
     a       102     1011
     b        13     1010001
     c        31     110011
     d        58     11010
     e       165     011
     f        26     101011
     g        27     110001
     h        80     0101
     i        65     11011
     k         3     110010101
     l        42     00101
     m        13     1100100
     n        76     0011
     o        93     1001
     p        15     001000
     q         1     1100101000
     r        79     0100
     s        44     10001
     t       124     000
     u        21     100000
     v        24     101001
     w        26     110000
     y        10     0010011

 6200 total Huffman-encoded bits
46.9% reduction
```

Codecheck URL:
http://codecheck.it/codecheck/files/1508010323ehsg28urjg6q8sjsian3rrpmi

Canvas: **Homework 10 Final** (there is no draft)
Due: Wednesday, August 5 at 11:59 PM